This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

Pelta Lumi — Annex B

Integración del Quantum Hash Framework (QHF) en Pelta Lumi

Especificación Técnica de Integración

El Quantum Hash Framework (QHF), desarrollado por Carlos Daniel Borrego Romero bajo licencia CC BY-NC-SA 4.0, define un descriptor determinístico de estado multi-escala que combina observables físicos (energía, entropía, tiempo, causalidad) con una capa criptográfica reproducible. Pelta Lumi adopta el QHF como mecanismo de sellado del Immutable Log Store, porque sus propiedades — determinismo, separación entre descriptor y primitiva criptográfica, y modularidad de la capa de hashing — permiten preparar el sistema para la migración post-cuántica sin cambiar el esquema de datos ni la arquitectura de detección.

1. Justificación de la Integración

1.1 Por qué el QHF es compatible con Pelta Lumi

El QHF define una función determinística:

$$H_q(O;t) := \mathcal{H}(X_O(t), S(\rho_O(t)), E_O, T(t), \mathcal{C}(t), \mathcal{M}_O)$$

donde:

- $X_O(t) = \mathrm{Tr}(\rho_O(t)\hat{H}_{\mathrm{eff}}(t))$: resumen escalar del estado del sistema
- $S(\rho_O(t))$: entropía de von Neumann (o surrogate clásico equivalente)
- E_O: escala de energía del sistema
- $T(t)$: índice temporal determinístico
- $\mathcal{C}(t)$: descriptor de consistencia causal
- $\mathcal{M}_O$: metadatos estructurados de provenance

Esta estructura es directamente aplicable a los eventos de telemetría de Pelta Lumi, donde cada evento de detección tiene un estado cuantificable, una medida de incertidumbre/entropía, una escala de

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

impacto, un timestamp determinístico, una relación de causalidad con eventos previos, y metadatos de provenance del sistema.

1.2 Separación descriptor-criptografía

El QHF separa explícitamente la etapa de construcción del descriptor de la etapa de representación criptográfica. Esto significa que la función de hash subyacente (SHA3-256, BLAKE3, o futura función post-cuántica) puede reemplazarse sin cambiar el modelo de datos de Pelta Lumi. Esta propiedad es crítica para la preparación cuántica.

1.3 Determinismo y reproducibilidad

La serialización canónica del QHF (JSON con claves ordenadas, precisión numérica fija, codificación UTF-8) garantiza que el mismo evento, con los mismos parámetros, siempre produce el mismo hash. Esto es esencial para la verificación de integridad del log inmutable de Pelta Lumi.

2. Mapeo de Dimensiones QHF a Eventos de Pelta Lumi

2.1 Tabla de Mapeo Operacional

Dimensión QHF	Símbolo	Indicador en Pelta Lumi	Método de Cálculo	Cobertura
State representation	$\rho_O(t)$	Vector de features normalizado del evento de telemetría	Normalización L1 del vector de features extraído por el Preprocessor	Alta
Effective observable summary	$X_O(t)$	Score de riesgo ponderado del evento	$X_O = \sum_i w_i \cdot f_i$ donde f_i son features normalizados y w_i pesos del modelo	Alta
Entropy descriptor	$S(\rho_O)$	Entropía de Shannon del vector de features normalizado	$S = -\sum_i p_i \log\ p_i$ donde p_i = componentes del vector de estado normalizado	Alta
Normalized entropy	S_{norm}	Entropía normalizada por clase de evento	$S_{\text{norm}} = S/S_{\max}$ donde $S_{\max} = \log(d)$ y d = dimensión del vector	Alta

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

Energy scale	E_O	Escala de impacto del evento (proxy de energía de procesamiento)	$E_O = k \cdot (f_{\text{cpu_delta}} + f_{\text{battery_delta}} + f_{\text{network_bytes}})$	Media
Temporal index	$T(t)$	Timestamp Unix en nanosegundos redondeado a intervalo fijo ($\Delta t = 1s$)	$T(t) = T_0 + n \cdot \Delta t,\ n \in \mathbb{Z},\ \Delta t > 0$	Alta
Causal descriptor	$\mathcal{C}(t)$	Código de admisibilidad causal (1 = causal, 0 = posible anomalía de orden)	Binary flag: $\mathcal{C} \in \{0,1\}$ basado en verificación de causalidad Granger	Alta
Structured metadata	$\mathcal{M}_O$	Metadata del evento: system_class, protocol, scale_class, model_version, device_class	Campos fijos con vocabulario controlado, serialización JSON canónica	Alta

2.2 Representación Continua del Descriptor QHF en Pelta Lumi

La representación continua es:

$$H_q^{\text{cont}}(O;t) = \text{Re}(H_q^{\text{cont}}) + i \cdot \text{Im}(H_q^{\text{cont}})$$

donde:

$$\text{Re}(H_q^{\text{cont}}) = \log_{10}(E_O) + \frac{\kappa}{100}(S_{\text{norm}} - S_{\text{ref}})$$

$$\text{Im}(H_q^{\text{cont}}) = \gamma \cdot S_{\text{norm}}$$

Parámetros por defecto para Pelta Lumi:

- $\kappa = 10.0$ (escala de corrección de entropía)
- $\gamma = 1.0$ (escala de la parte imaginaria)
- $S_{\text{ref}} = 0.5$ (punto de referencia de entropía normalizada)
- $S_{\max} = \log(d)$ donde d es la dimensión del vector de features del evento

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

- Serialización: canonical JSON con claves ordenadas alfabéticamente, UTF-8, float con 8 decimales

2.3 Representación Discreta (Hash Criptográfico)

$$h_q(O;t) := \text{SHA3-256}(\text{Serialize}(\Xi_{PL}(t)))$$

donde el tuple canónico es:

$$\Xi_{PL}(t) := (H_q^{\text{cont}}(O;t), T(t), \mathcal{C}(t), \mathcal{M}_O)$$

3. Pipeline de Implementación On-Device

3.1 Flujo de Cálculo del QHF Hash (Pelta Lumi Event)

```
# Pseudocódigo de referencia — implementación on-device en Kotlin/Swift
def compute_qhf_hash(event: PeltaLumiEvent, params: QHFParams) -> QHFBlock:

  # Paso 1: Construir vector de estado normalizado
  feature_vector = event.features  # ya normalizado por Preprocessor
  rho = normalize_state_vector(feature_vector)  # suma = 1, no negativo

  # Paso 2: Calcular entropía de Shannon como surrogate de S(rho)
  S = -sum(p * log(p) for p in rho if p > 0)
  S_max = log(len(rho))  # S_max = log(d) para sistema de dim d
  S_norm = S / S_max if S_max > 0 else 0.0

  # Paso 3: Score de riesgo ponderado como X_O
  X_O = compute_weighted_score(rho, event.model_weights)

  # Paso 4: Escala de energía como proxy de impacto
  E_O = params.k_scale * (event.cpu_delta + event.battery_delta +
                event.network_bytes / params.bytes_normalizer)
  E_O = max(E_O, 1e-10)  # garantizar E_O > 0 para log10

  # Paso 5: Índice temporal determinístico
  T = round(event.timestamp_ns / params.delta_t_ns) * params.delta_t_ns
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

```
# Paso 6: Descriptor causal
C = 1 if event.causal_order_verified else 0

# Paso 7: Metadatos estructurados
M = {
   "system_class": "mobile_telemetry",
   "protocol": "pelta_lumi_v5",
   "scale_class": event.event_type,
   "model_version": params.model_version,
   "device_class": params.device_class
}

# Paso 8: Representación continua
correction = params.kappa * (S_norm - params.S_ref) / 100.0
Re = log10(E_O) + correction
Im = params.gamma * S_norm
H_cont = complex(Re, Im)

# Paso 9: Serialización canónica
canonical_tuple = {
   "H_cont_Re": round(Re, 8),
   "H_cont_Im": round(Im, 8),
   "T_index": T,
   "C_code": C,
   "metadata": dict(sorted(M.items()))  # claves ordenadas
}
serialized = json.dumps(canonical_tuple, sort_keys=True,
                        ensure_ascii=True, separators=(',', ':'))

# Paso 10: Hash criptográfico
qhf_hash = sha3_256(serialized.encode('utf-8')).hexdigest()

return QHFBlock(
   X_O=X_O, S_norm=S_norm, E_O_log10=log10(E_O),
   T_index=T, C_code=C, metadata=M,
   H_cont=H_cont, qhf_hash=qhf_hash,
   params=params
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

```
  )
```

3.2 Parámetros de Configuración del QHF en Pelta Lumi

```
{
  "qhf_config": {
    "hash_function": "SHA3-256",
    "hash_function_pqc_planned": "BLAKE3",
    "kappa": 10.0,
    "gamma": 1.0,
    "S_ref": 0.5,
    "S_max_rule": "log_d",
    "delta_t_ns": 1000000000,
    "k_scale": 1.0,
    "bytes_normalizer": 1048576,
    "serialization": "canonical_json_sorted_keys_utf8_float8",
    "causal_coding_rule": "binary_0_1",
    "energy_floor": 1e-10,
    "version": "1.0.0"
  }
}
```

4. Preparación Post-Cuántica

4.1 Amenaza Cuántica a la Criptografía Actual

Los algoritmos de firma actuales usados en Pelta Lumi (ECDSA-P256) y las funciones de hash (SHA-256) son vulnerables ante computadores cuánticos con suficiente potencia mediante:

- **Algoritmo de Shor:** rompe ECDSA y RSA en tiempo polinómico.
- **Algoritmo de Grover:** reduce la seguridad efectiva de SHA-256 de 256 bits a 128 bits equivalentes.

El NIST publicó en 2024 los estándares FIPS 203 (CRYSTALS-Kyber para KEM), FIPS 204 (CRYSTALS-Dilithium para firmas digitales) y FIPS 205 (SPHINCS+ para firmas basadas en hash).

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

4.2 Arquitectura de Migración Post-Cuántica de Pelta Lumi

La separación que el QHF establece entre la etapa de construcción del descriptor y la etapa de hashing permite una migración limpia sin cambio de esquema:

Componente	Hoy 2026	Post-cuántico (v6.0+)	Impacto en esquema
Hash de eventos QHF	SHA3-256	BLAKE3 o SHA3-512	Solo cambio de parámetro `hash_function`
Firma del log block	ECDSA-P256	CRYSTALS-Dilithium-3 (FIPS 204)	Solo cambio de parámetro `signature.algorithm`
Firma del modelo AI	ECDSA-P256	CRYSTALS-Dilithium-3	Solo cambio de parámetro en Model Registry
KEM para telemetría edge	TLS 1.3 (X25519)	TLS 1.3 + CRYSTALS-Kyber (FIPS 203)	Upgrade de TLS stack
Merkle root anchor	SHA-256	BLAKE3	Solo cambio de primitiva en WORM

4.3 Ruta de Migración por Fases

Fase 1 — Preparación (ahora 2026):

- Implementar QHF como descriptor de estado con SHA3-256 y ECDSA-P256.
- Configurar el campo `pqc_ready: true` y `pqc_algorithm_planned` en cada bloque.
- Abstraer la capa criptográfica detrás de interfaces intercambiables (`CryptoProvider`, `HashProvider`).
- Documentar todos los puntos de uso de primitivas clásicas.

Fase 2 — Híbrido (v5.5, disponibilidad de librerías estables):

- Implementar firmas híbridas: ECDSA-P256 + CRYSTALS-Dilithium-3 en paralelo.
- Cada bloque de log incluye ambas firmas; verificadores pueden usar cualquiera.
- Objetivo: sin ruptura de compatibilidad hacia atrás.

Fase 3 — Migración completa (v6.0, cuando NIST PQC sea estándar maduro):

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

- Reemplazar ECDSA-P256 con CRYSTALS-Dilithium-3 como firma principal.
- Reemplazar SHA3-256 con BLAKE3 para hashing de eventos (más eficiente en móviles).
- Mantener SHA3-256 como fallback para verificación de logs históricos.

4.4 Compatibilidad de BLAKE3 con el QHF

El QHF establece explícitamente que "el framework no está intrínsecamente ligado a SHA3-256 y puede implementarse con diferentes primitivas criptográficas determinísticas, incluyendo SHA-2, SHA-3, BLAKE2, BLAKE3, o capas de hashing post-cuántico compatibles, sin cambiar la estructura del modelo de alto nivel."

BLAKE3 es especialmente adecuado para Pelta Lumi en dispositivos móviles porque:

- Rendimiento ~6x superior a SHA3-256 en hardware de 64-bit (ARM).
- Diseñado para implementaciones en tiempo real con latencia crítica.
- Resistencia extendida a Grover: configuración de 256-bit ofrece 256 bits de seguridad clásica y 128 bits post-cuánticos.
- Adoptado por proyectos de infraestructura crítica (Cloudflare, Rust ecosystem).

5. Integración con Sistemas Actuales

5.1 API REST de Exportación con QHF

```
GET /api/v1/log/export?format=json-ld&qhf=true
Authorization: Bearer {user_token}

Response 200:
{
 "export_id": "uuid-v4",
 "generated_at": "ISO8601",
 "qhf_export_root": "sha3-256:abc...",
 "qhf_params": { ... },
 "blocks": [ { ...bloque QHF... }, ... ],
 "merkle_root": "sha3-256:xyz...",
 "signature": { "algorithm": "ECDSA-P256", "value": "..." },
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

```
 "verification_wasm": "https://verify.pelta-lumi.io/v5/verify.wasm"
}
```

5.2 Exportación a SIEM (CEF con QHF Hash)

Cada evento de detección se transforma a Common Event Format (CEF) para integración con Splunk, QRadar y Microsoft Sentinel, incluyendo el QHF hash como campo de referencia:

```
CEF:0|PeltaLumi|Agent|5.0|INFLUENCE_DETECTED|Geofence nudge detectado|7|
 rt=2026-05-14T20:00:00Z
 src=device_hash:sha256:abc123
 app=ShopApp:hash:def456
 msg=Geofence trigger + beacon BLE + dark pattern temporal pressure
 cs1=INFERENCE_STRONG cs1Label=certaintyLevel
 cs2=0.83 cs2Label=confidenceScore
 cs3=sha3-256:qhf789abc cs3Label=qhfBlockHash
 cs4=CRYSTALS-Dilithium-3 cs4Label=signatureAlgorithmPlanned
```

5.3 Integración OpenTelemetry

Pelta Lumi expone un exportador OTLP que envía trazas de detección al backend de observabilidad del operador (Grafana, Jaeger, Datadog), con el QHF hash como atributo de traza para correlación:

```
{
 "resourceSpans": [{
  "resource": {
   "attributes": [
    {"key": "service.name", "value": {"stringValue": "pelta-lumi-agent"}},
    {"key": "service.version", "value": {"stringValue": "5.0.0"}}
   ]
  },
  "scopeSpans": [{
   "spans": [{
    "traceId": "...", "spanId": "...",
    "name": "detection.real_time_influence",
    "attributes": [
     {"key": "pelta.certainty", "value": {"stringValue": "INFERENCE_STRONG"}},
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full “License and Conditions of Use” notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

```
      {"key": "pelta.qhf_hash", "value": {"stringValue": "sha3-256:qhf789abc"}},
      {"key": "pelta.confidence", "value": {"doubleValue": 0.83}}
     ]
    }]
   }]
  }]
}
```

6. Verificación Portable del Log QHF

6.1 Módulo WASM de Verificación

El log exportado por Pelta Lumi puede verificarse en cualquier entorno que soporte WebAssembly, sin necesidad de instalar Pelta Lumi. El módulo `pelta-lumi-verify.wasm` implementa:

1. Deserialización del export JSON-LD.
2. Verificación de la cadena de QHF hashes (prev_hash → hash_event → merkle_root).
3. Verificación de la firma criptográfica del export root.
4. Verificación opcional de anclaje RFC 3161 (si el export incluye timestamp TSA).
5. Salida: `VALID | TAMPERED | SIGNATURE_INVALID | CHAIN_BROKEN` con detalle del bloque afectado.

6.2 Proceso de Verificación

```
Input: export JSON-LD + certificado de clave pública del dispositivo
   ↓
1. Parsear bloques y extraer chain de hashes QHF
   ↓
2. Para cada bloque N:
  a. Re-calcular QHF hash del evento con los parámetros del bloque
  b. Verificar que event_hash == QHF hash calculado
  c. Verificar que prev_hash == hash del bloque N-1
   ↓
3. Verificar Merkle root del conjunto de bloques
   ↓
4. Verificar firma ECDSA-P256 (o Dilithium en v6.0+) del export root
   ↓
```

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

```
5. Si anchor RFC 3161 presente: verificar timestamp externo
   ↓
Output: VALID (todos los bloques íntegros) | TAMPERED (indicar bloque afectado)
```

7. Consideraciones de Privacidad en el QHF

7.1 Qué se incluye en el QHF hash

El descriptor QHF de cada bloque contiene únicamente:

- Variables escalares derivadas (X_O, S_norm, E_O_log10, T, C) — sin datos de usuario identificables.
- Metadatos de sistema (tipo de evento, versión, clase de dispositivo) — sin identificadores directos.
- Hashes de identificadores de apps y dominios (SHA-256) — no reversibles.

7.2 Qué NO se incluye en el QHF hash

- Identificadores de usuario o dispositivo en texto claro.
- Contenido de notificaciones push (solo análisis de features, no texto crudo).
- Payload de tráfico de red (solo metadatos).
- Datos de ubicación precisos (solo zona de geofence, no coordenadas).

7.3 Cumplimiento con la licencia QHF (CC BY-NC-SA 4.0)

La implementación de Pelta Lumi del QHF cumple con todos los términos de la licencia:

- **Atribución:** Carlos Daniel Borrego Romero es mencionado como autor del QHF en toda documentación y en el campo `qhf_attribution` del export.
- **No comercial:** el QHF se usa exclusivamente para protección del usuario, no para fines comerciales de perfilamiento.
- **ShareAlike:** toda mejora al módulo QHF de Pelta Lumi se publica bajo la misma licencia CC BY-NC-SA 4.0.

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

- **Sin datos de usuarios para entrenamiento comercial:** los logs QHF no se usan para entrenar sistemas comerciales de terceros.

8. Conclusión del Annex B

La integración del Quantum Hash Framework en Pelta Lumi aporta cuatro beneficios concretos y medibles:

1. **Determinismo verificable:** el mismo evento siempre produce el mismo QHF hash, haciendo el log completamente reproducible y auditable.
2. **Preparación post-cuántica:** la separación descriptor-criptografía del QHF permite reemplazar SHA3-256/ECDSA por BLAKE3/CRYSTALS-Dilithium sin cambiar el esquema de datos.
3. **Interoperabilidad:** el export JSON-LD con QHF hash es verificable en cualquier entorno que soporte WebAssembly, independientemente de la plataforma donde se generó.
4. **Evidencia legalmente admisible:** la cadena de QHF hashes con firma criptográfica y anclaje RFC 3161 opcional cumple los requisitos de admisibilidad de evidencia digital en los marcos GDPR, LGPD, CCPA y DSA.

LICENSE AND CONDITIONS OF USE.
The content is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) license:
https://creativecommons.org/licenses/by-nc-sa/4.0/
By using this content and model, you agree to:
Provide appropriate credit, include a link to the above license, and indicate if changes were made.
Not use the material or any derivative works for commercial purposes.
Distribute any adaptations or derivative works only under the same CC BY-NC-SA 4.0 license.
Not apply legal terms or technological measures that legally restrict others from doing anything the license permits.
Data logging and trade secrets policy.
The design, implementation, and deployment of this model, as well as any derivatives, do not authorize the collection, storage, or exploitation of IP addresses, unique device identifiers, or other technical traceability data for commercial profiling, targeted advertising, or any other economic exploitation.
Any system implementing this model must limit the logging and retention of IP addresses and other identifiable data strictly to what is necessary to:
(a) maintain technical security and service integrity; and
(b) comply with applicable legal obligations (for example, any minimum log retention required by law).
It is prohibited to use this model or its derivatives to:
capture, store, or exploit trade secrets, proprietary know-how, or confidential information of users or third parties for any purpose other than providing the technical service described in this documentation;

This model and documentation are provided for non-commercial use only under the CC BY-NC-SA 4.0 license.
No commercial use, profiling, or exploitation of IP logs or trade secrets is permitted.
See full "License and Conditions of Use" notice for details.
In case of conflict or ambiguity, the most restrictive interpretation in favor of non-commercial use, user privacy, and protection of trade secrets shall prevail.
Author: Carlos Daniel Borrego Romero

or train or improve closed, commercial systems using data that contains confidential information of third parties without their prior, explicit, written consent.

---+EOD+---